Network Working Group Larry Garlick
Request for Comments 721 SRI-ARC
NIC 36636 1 September 76

Out-of-Band Control Signals
in a
Host-to-Host Protocol

This note addresses the problem of implementing a reliable out-of-band signal for use in a host-to-host protocol. It is motivated by the fact that such a satisfactory mechanism does not exist in the Transmission Control Protocol (TCP) of Cerf et. al. [reference 4, 6] In addition to discussing some requirements for such an out-of-band signal (interrupts) and the implications for the implementation of the requirements, a discussion of the problem for the TCP case will be presented.

While the ARPANET host-to-host protocol does not support reliable transmission of either data or controls, it does meet the other requirements we have for an out-of-band control signal and will be drawn upon to provide a solution for the TCP case.

The TCP currently handles all data and controls on the same logical channel. To achieve reliable transmission, it provides positive acknowledgement and retransmission of all data and most controls. Since interrupts are on the same channel as data, the TCP must flush data whenever an interrupt is sent so as not to be subject to flow control.

Functional Requirements

It is desirable to insure reliable delivery of an interrupt. The sender must be assured that one and only one interrupt is delivered at the destination for each interrupt it sends. The protocol need not be concerned about the order of delivery of interrupts to the user.

The interrupt signal must be independent of data flow control mechanisms. An interrupt must be delivered whether or not there are buffers provided for data, whether or not other controls are being handled. The interrupt should not interfere with the reliable delivery of other data and controls.

The host-to-host protocol need not provide synchronization between the interrupt channel and the data-control channel. In fact, if coupling of the channels relies on the advancement of sequence numbers on the data-control channel, then the interrupt channel is no longer independent of flow control as required above. The synchronization with the data stream can be performed by the user by

marking the data stream when an interrupt is generated. The interrupt need not be coupled with other controls since it in no way affects the state of a connection.

Once the interrupt has been delivered to the user, no other semantics are associated with it at the host-to-host level.

Implications

To provide for reliable delivery and accountability of interrupt delivery, an acknowledgement scheme is required. To associate interrupt acknowledgements with the correct interrupt, some naming convention for interrupts is necessary. Sequence numbers provide such a naming convention, along with the potential for providing an ordering mechanism.

A separate interrupt channel is required to make interrupts independent of flow control. A separate sequence number space for naming interrupts is also necessary. If the sequence numbers are from the same sequence number space as some other channel, then sending an interrupt can be blocked by the need to resynchronize the sequence numbers on that channel.

In the current TCP, which uses one channel for data, controls, and interrupts, flushing of data is combined with the interrupt to bypass the flow control mechanism. However, flushing of resynchronization controls is not allowed and receipt of these controls is dependent on the acknowledgement of all previous data. The ARPANET protocol, while not providing for reliable transmission, does provide for the separation of the interrupt-control channel and the data channel.

Multiple Channels and Sequence Numbers

If multiple channels are to be used for a connection, then it becomes interesting to determine how the sequence numbers of the channels can be coupled so that sequence number maintenance can be done efficiently.

Assigning sequence numbers to each octet of data and control, as in the TCP, allows positive acknowledgement and ordering. However, since packets are retransmitted on timeout, and since multi-path packet switch networks can cause a packet to stay around for a long time, the presence of duplicate packets and out-of-order packets must be accounted for. A sequence number acceptability test must be performed on each packet received to determine if one of the following actions should be taken:

Acknowledge the packet and pass it on to the user.

Acknowledge the packet, but do not send it to the user, since it has already been delivered.

Discard the packet; the sequence number is not believable.

Acceptability on Channel 0

To determine the action to be taken for a packet, acceptability ranges are defined. The following three ranges are mutually exclusive and collectively exhaustive of the sequence number space (see Figure 1):

Ack-deliver range (ADR)

Ack-only range (AOR)

Discard range (DR)

```
              ACCEPTABILITY RANGES

    DR       AOR             ADR             DR
\\=====)[===========)[==================](========\\
        ^           ^^                  ^^
        !           !\                  !\
        !           ! FCLE              ! DRLE
      AOLE       AORE                ADRE
```

Figure 1

Let S = size of sequence number space (number per octet)

x = sequence number to be tested

FCLE = flow control left window edge

ADRE = (FCLE+ADR) mod S = Ack-deliver right edge (Discard left edge - 1)

AOLE = (FCLE-AOR) mod S = Ack-only left edge (Discard right edge + 1)

TSE = time since connection establishment (in sec)

MPL = maximum packet lifetime (in sec)

TB = TCP bandwidth (in octets/sec)

For any sequence number, x, and packet text length, l, if

(AOLE <= x <= ADRE) mod S and

(AOLE <= x+l-1 <= ADRE) mod S

then the packet should be acknowledged.

If x and l satisfy

(FCLE <= x <= ADRE) mod S and

(FCLE <= x+l-1 <= ADRE) mod S

then x can also be delivered to the user; however, ordered delivery requires that x = FCLE.

A packet is not in a range only if all of it lies outside a range. When a packet falls in more than one range, precedence is ADR, then AOR, then DR. When a packet falls in the AOR then an ACK should be sent, even if a packet has to be created. The ACK will specify the current left window edge. This assures acknowledgment of all duplicates.

ADRE is exactly the maximum sequence number ever "advertised" through the flow control window, plus one. This allows for controls to be accepted even though permission for them may never have been explicitly given. Of course, each time a control with a sequence number equal to the ADRE is sent, the ADRE must be incremented by one.

AOR is set so that old duplicates (from previous incarnations of the connection) can be detected and discarded. Thus

AOR = Min(TSE, MPL) * TB

Synchronization and Resynchronization Problems

A special problem arises concerning detection of packets (old duplicates) in the network that have sequence numbers assigned by old instances of a connection. To handle this reliably, careful selection of the initial sequence number is required [ref. 2, 3] as well as periodic checks to determine if resynchronization of sequence numbers is necessary. The overhead of such elaborate machinery is expensive and repeating it for each additional channel is undesirable.

Acceptability on Channel i

We have concluded that the only savings realizable in the muiltple channel case is to use channel zero's initial sequence number and resynchronization maintenance mechanism for the additional channels. This can be accomplished by coupling each additional channel to channel zero's sequence numbers (CZSN), so that each item on channel i carries a pair of sequence numbers, the current CZSN and the current channel i's sequence number (CISN).

The acceptablility test of items on channel i is a composite test of both sequence numbers. First the CZSN is checked to see if it would be acknowledged if it were an octet received on channel zero. Only if it would have been discarded would the item on channel i be discarded. Having passed the CZSN test, the CISN is checked to see if the item is deliverable and acknowledgable with respect to the CISN sequence number space. The CISN test is a check for everything but the existence of old duplicates from old instances of the connection and is performed like the check for channel zero items.

It has been shown that to implement additional channels for a TCP connection, two alternatives are available-- (1) provide each channel with its own initial sequence number and resynchronization maintenance mechanism or (2) provide one initial sequence number and resynchronization maintenance mechanism for all channels through channel zero's mechanism. It is hard for us to compare the two alternatives, since we have no experience implementing any resynchronization maintenance mechanism.

TCP Case

To implement a completely reliable separate interrupt channel for TCP requires a channel with a full sequence number space. It is possible to compromise here and make the interrupt number space smaller than that required to support consumption of numbers at the TCP's

bandwidth. What is lost is the total independence of the flow control from the data-control channel. Normally, the data-control sequence numbers will change often enough so that wraparound in the interrupt number space causes no problems.

Things become slightly messy when many interrupts are generated in quick succession. Even if the interrupt numbers are acknowledged, they cannot be reused if they refer to the same data-control sequence number, until a full packet lifetime has elapsed. This can be remedied in all but one case by sending a NOP on the data-control channel so that the next set of interrupts can refer to a new data-control sequence number. However, if the data-control channel is blocked due to flow control and a resynchronizing control (DSN in the TCP case) was just sent, a NOP cannot be created until the resynchronization is complete and a new starting sequence number is chosen. Thus to send another interrupt, the TCP must wait for a packet lifetime or an indication that it can send a NOP on the data-control channel. (In reality, a connection would probably be closed long before a packet lifetime elapsed if the sender is not accepting data from the receiver. [reference 1])

REFERENCES

(1) J. Postel, L. Garlick, R. Rom, "TCP Specification (AUTODIN II)," ARC Catalog #35938, July 1976.

(2) R. Tomlinson, "Selecting Sequence Numbers," INWG Protocol Note #2, September 1974.

(3) Y. Dalal, "More on Selecting Sequence Numbers," INWG Protocol Note #4, October 1974.

(4) V. Cerf, Y. Dalal, C. Sunshine, "Specification of Internet Transmission Control Program," INWG General Note #72, December 1974 (Revised). [Also as RFC 675, NIC Catalog #31505.]

(5) Cerf, V., "TCP Resynchronization," SU-DSL Technical Note #79, January 1976.

(6) Cerf, V. and R. Kahn, "A Protocol for Packet Network Intercommunication," IEEE Transactions on Communication, Vol COM-20, No. 5, May 1974.

(7) C. Sunshine, "Interprocess Communication Protocols for Computer Networks," Digital Systems Laboratory Technical Note #105, December 1975.